Cross Platform GUI Programming with Python

Elliot Garbus
Seth Abraham

DesertPy - November 28, 2018

Introduction & Agenda

Your Presenters:

- Elliot Garbus - Management Consultant & Python Enthusiast
- Seth Abraham - Professor of Electrical Engineering & Python Enthusiast

Agenda:

- Choosing a GUI Framework
- The Projects
- Kivy: Key Concepts and Demonstrations
- Applying the Concepts to the Projects
- Resources

Choosing a GUI framework

- Lots of options, not a lot of guidance…
- Considered:
  - tkinter - a standard python lib, based on TCL/TK
  - WxPython - based on WxWidgets
  - PyQt or Pyside (?) based of Qt
  - Kivy - Python native, OpenGL ES accelerated
  - Beeware Toga - Python + Native Widgets
  - Python + Electron - node.js + css + chromium

The Criteria

Target Design: Editor for a piece of music hardware

Cross Platform: MacOS and Windows required, mobile nice to have

Look: Modern look, custom widgets, Native not required

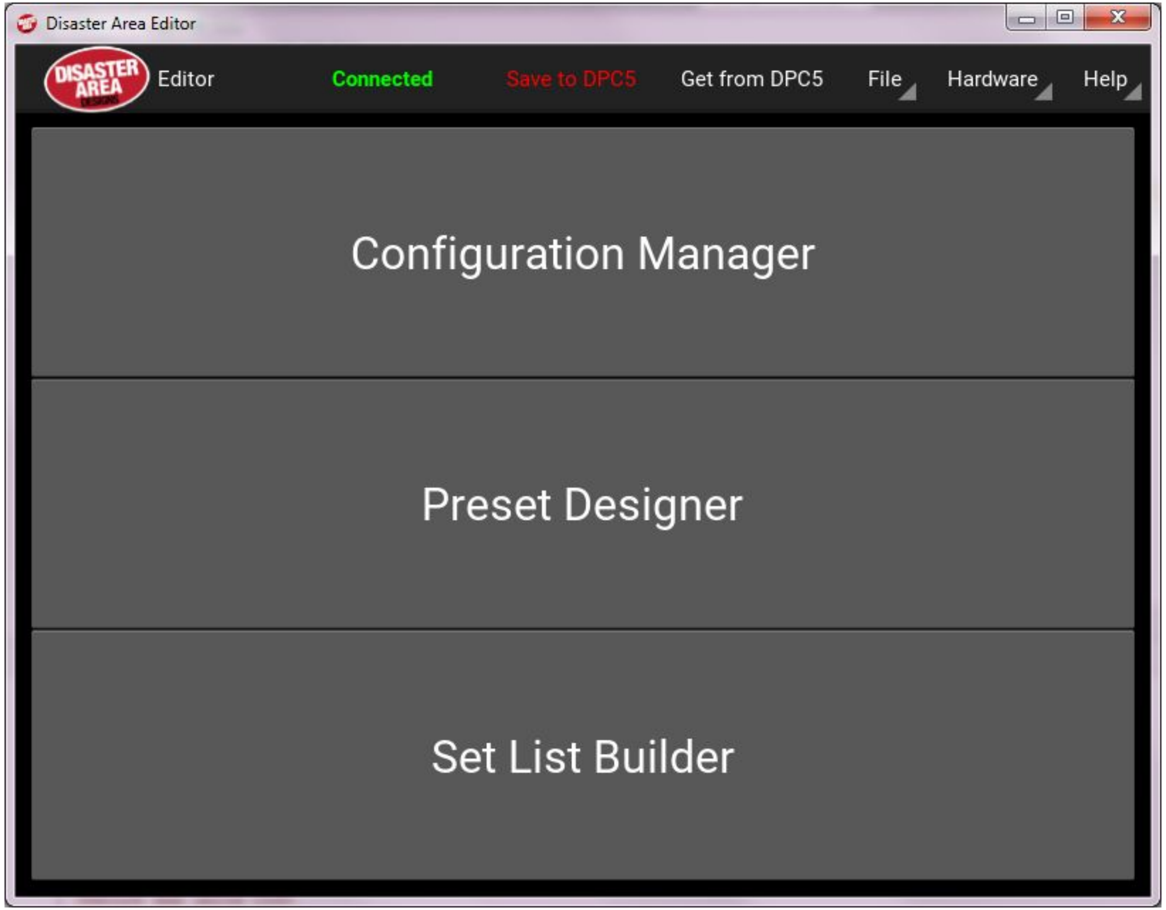
Evaluation:

Toolkit	Pro	Con
tkinter	Native Look with ttk widgets Ships with Python	Can’t draw an anti-aliased curve (UGLY) Weak documentation
WxPython	Native Look	Weak documentation, No Mobile
Qt	Mature, Commercial product, Cross platform	Dual license PySide/PyQT confusion (resolved, not released)
Kivy ✓	Modern look, Cross platform, Well supported, Excellent Documentation, MIT License (Free)	Not a native look
Toga	Python Native, Native look	Immature, early stage development
Electron	Modern Web Technologies as a GUI	Very big distributable (+70M to 200M) Complexity

The Projects(1/2): Midi Controller

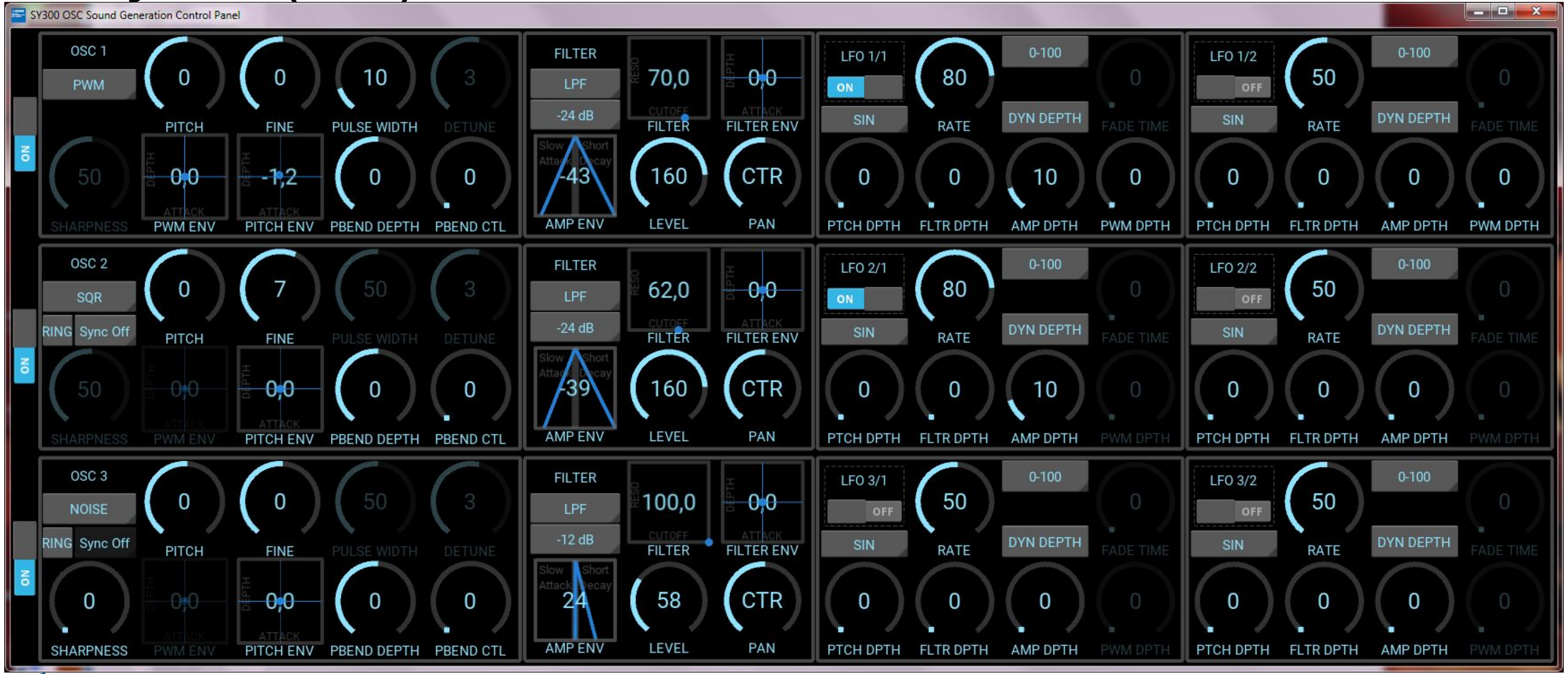
Configuration software for a MIDI foot controller

Used to connect and control multiple effects devices



*This software is not a Disaster Area Product

The Projects(2/2):



SY300 OSC Sound Generation Control Panel

OSC 1	ON	PWM	PITCH 0	FINE 0	PULSE WIDTH 10	DETUNE 3	SHARPNESS 50	PWM ENV 0,0	PITCH ENV -1,2
			PBEND DEPTH 0	PBEND CTL 0					
FILTER	LPF	-24 dB	FILTER 70,0	FILTER ENV 0,0	AMP ENV -43	LEVEL 160	PAN CTR		
LFO 1/1	ON	SIN	RATE 80	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 10
			PWM DPTH 0						
LFO 1/2	OFF	SIN	RATE 50	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 0
			PWM DPTH 0						
OSC 2	ON	SQR	RING	Sync Off	PITCH 0	FINE 7	PULSE WIDTH 50	DETUNE 3	SHARPNESS 50
			PWM ENV 0,0	PITCH ENV 0,0	PBEND DEPTH 0	PBEND CTL 0			
FILTER	LPF	-24 dB	FILTER 62,0	FILTER ENV 0,0	AMP ENV -39	LEVEL 160	PAN CTR		
LFO 2/1	ON	SIN	RATE 80	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 10
			PWM DPTH 0						
LFO 2/2	OFF	SIN	RATE 50	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 0
			PWM DPTH 0						
OSC 3	ON	NOISE	RING	Sync Off	PITCH 0	FINE 0	PULSE WIDTH 50	DETUNE 3	SHARPNESS 0
			PWM ENV 0,0	PITCH ENV 0,0	PBEND DEPTH 0	PBEND CTL 0			
FILTER	LPF	-12 dB	FILTER 100,0	FILTER ENV 0,0	AMP ENV 24	LEVEL 58	PAN CTR		
LFO 3/1	OFF	SIN	RATE 50	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 0
			PWM DPTH 0						
LFO 3/2	OFF	SIN	RATE 50	0-100	DYN DEPTH	FADE TIME 0	PTCH DPTH 0	FLTR DPTH 0	AMP DPTH 0
			PWM DPTH 0						

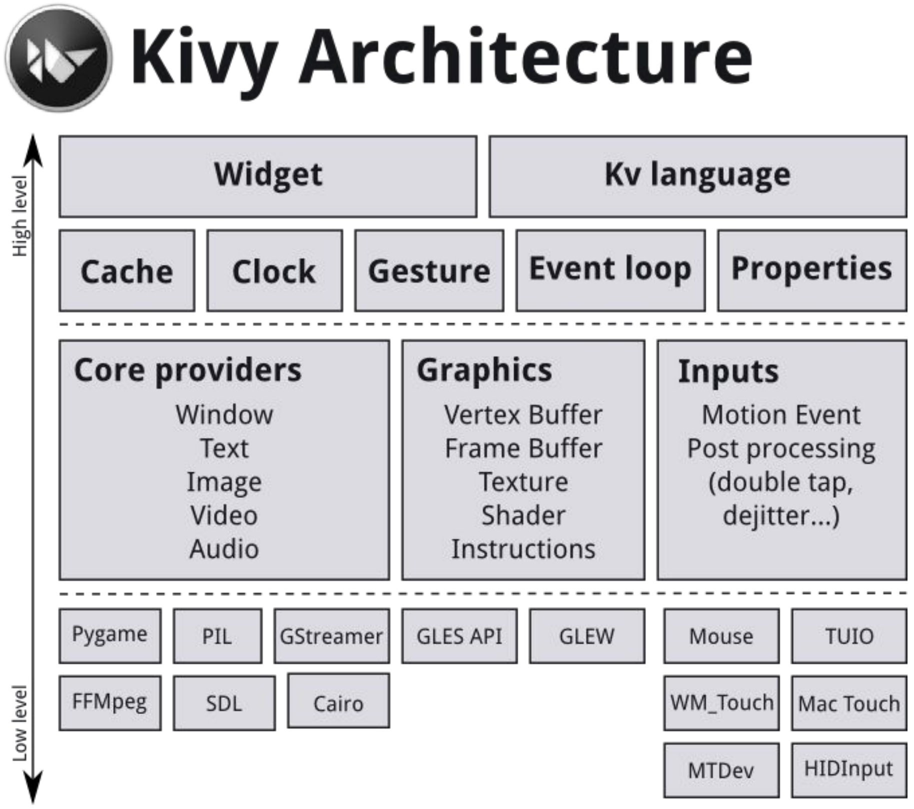
Control Panel for a Synthesizer
Used to configure sound generation elements

BOSS GUITAR SYNTHESIZER SY-300

*This software is not a Boss Product

Kivy

- **Fresh** Created for multi-touch & Python
- **Fast** Development and Execution
- **Flexible** Win, MacOS, Linux, OS X, Android, Raspberry Pi...
- **Funded** Professionally Developed
- **Free** to use, Even for commercial projects

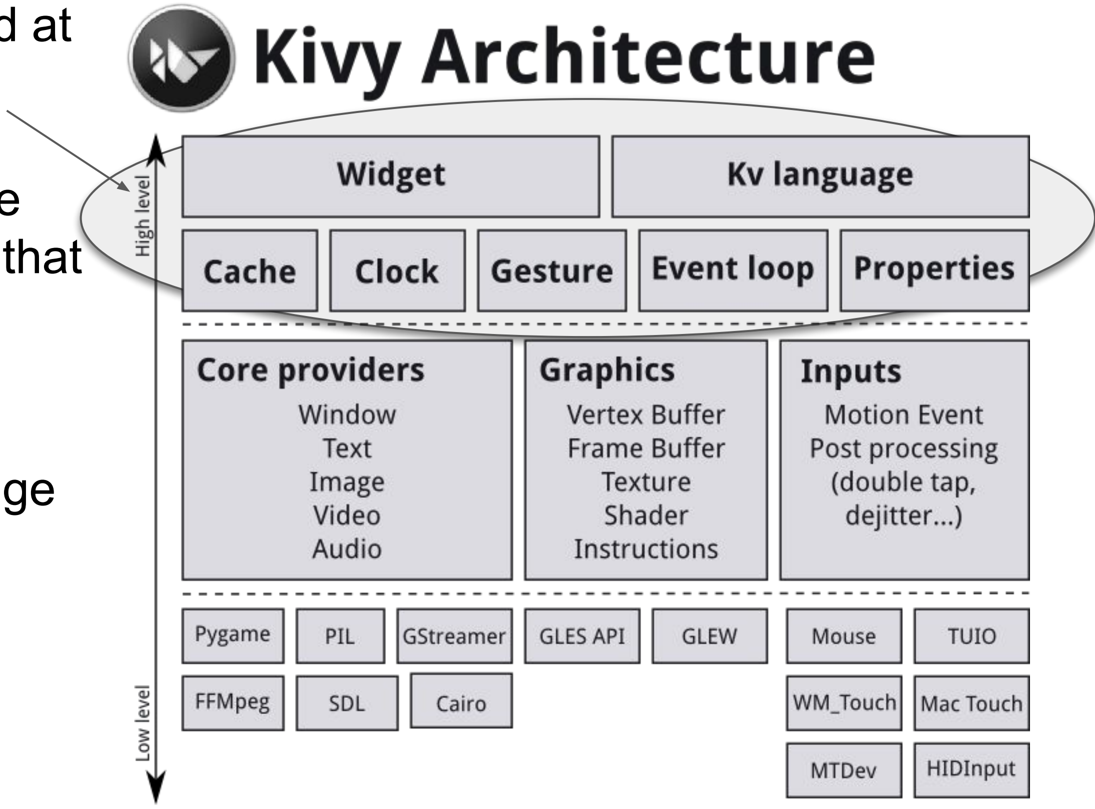


Kivy: 5 Key Concepts

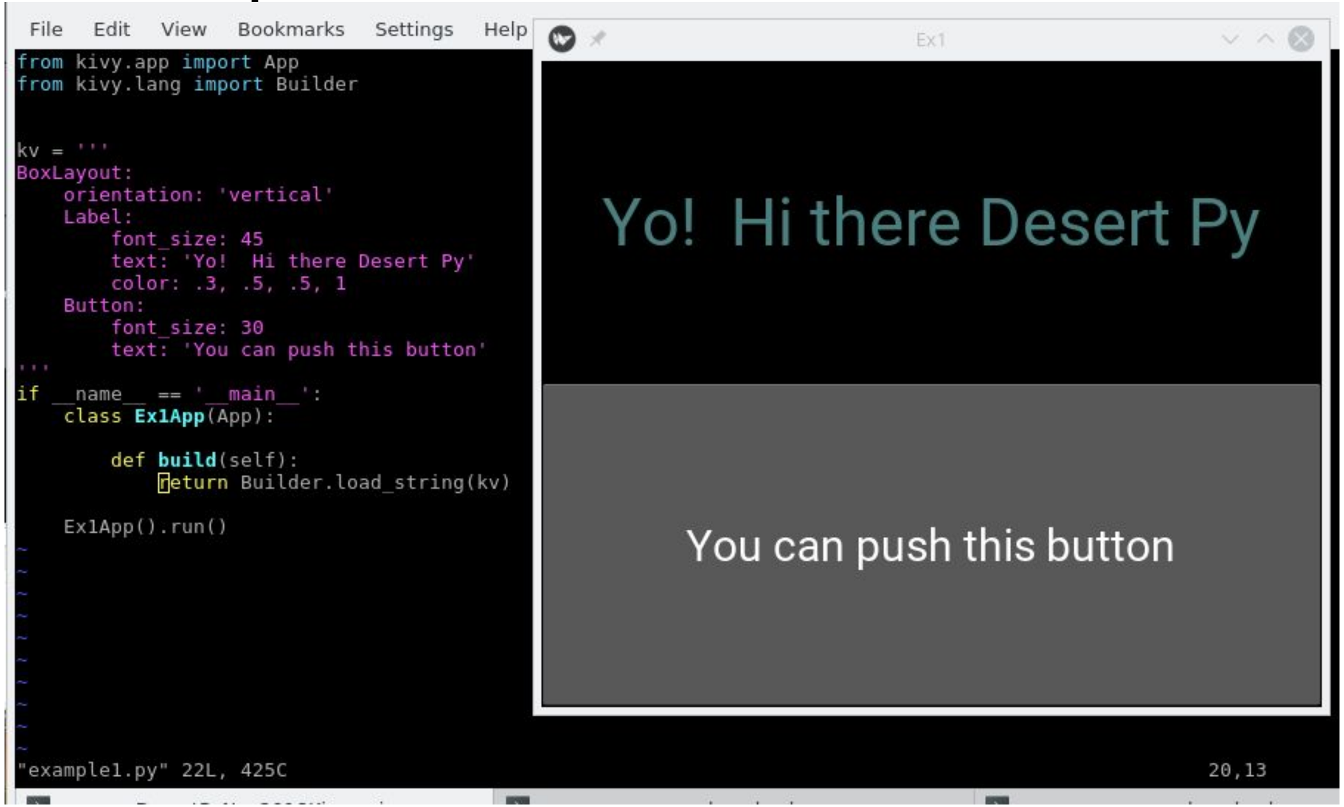
Application Development is focused at the top of the framework stack

Kv Language - Separates Interface design from app logic. An ‘outline’ that defines arrangement and simple behaviors

Layouts - containers used to arrange and size widgets



Example 1 -- Hello World

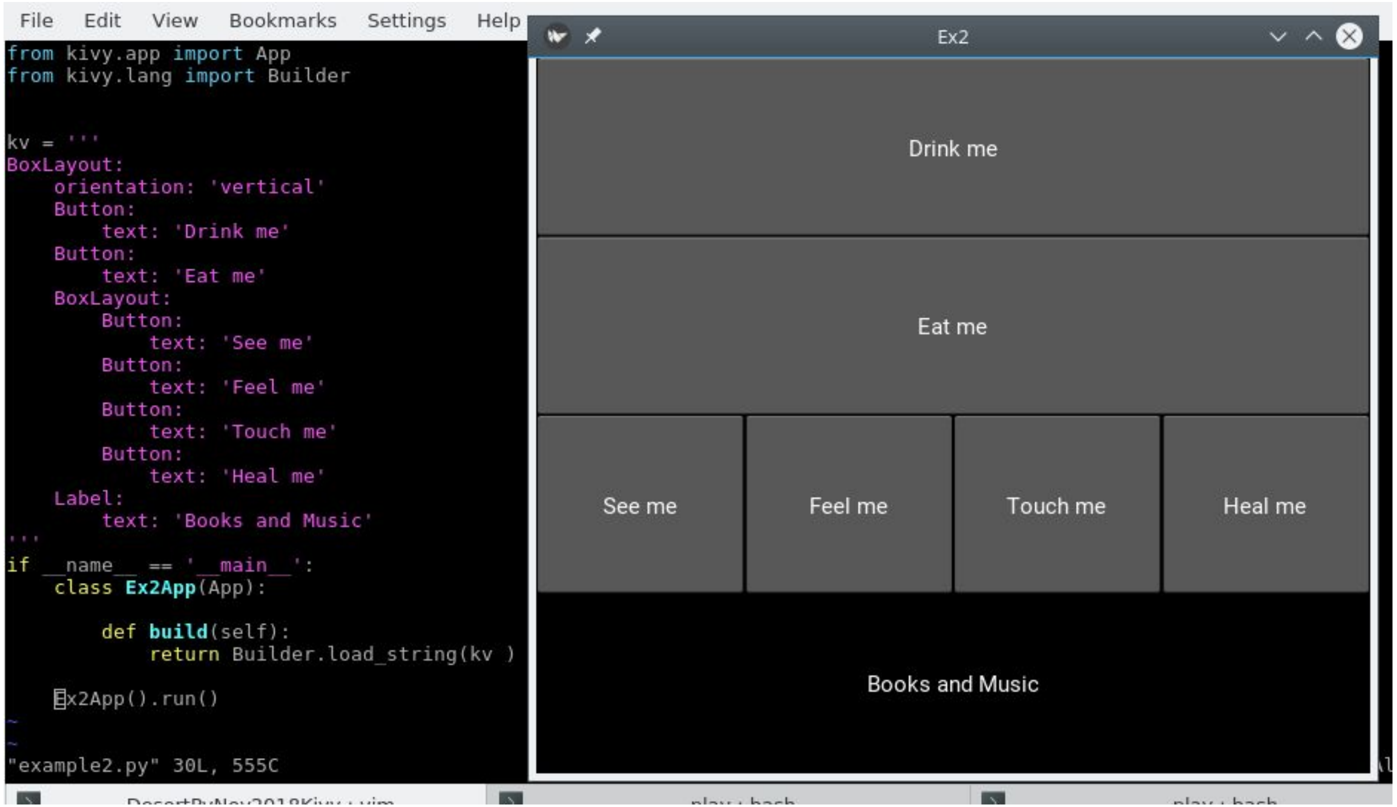


```
from kivy.app import App
from kivy.lang import Builder


kv = '''
BoxLayout:
    orientation: 'vertical'
    Label:
        font_size: 45
        text: 'Yo!  Hi there Desert Py'
        color: .3, .5, .5, 1
    Button:
        font_size: 30
        text: 'You can push this button'
'''
if __name__ == '__main__':
    class Ex1App(App):

        def build(self):
            return Builder.load_string(kv)

    Ex1App().run()
~
~
~
~
~
~
~
~
~
~
"example1.py" 22L, 425C                                          20,13
```

- Simple to create
  - COMPLETE code shown!
  - Execute with python3
  - Code on github
- One Box layout
- A Label
  - Text added,
  - Font & color change
- A Button
  - Can be pressed

https://github.com/sabrah12/DesertPyNov2018Kivy.git

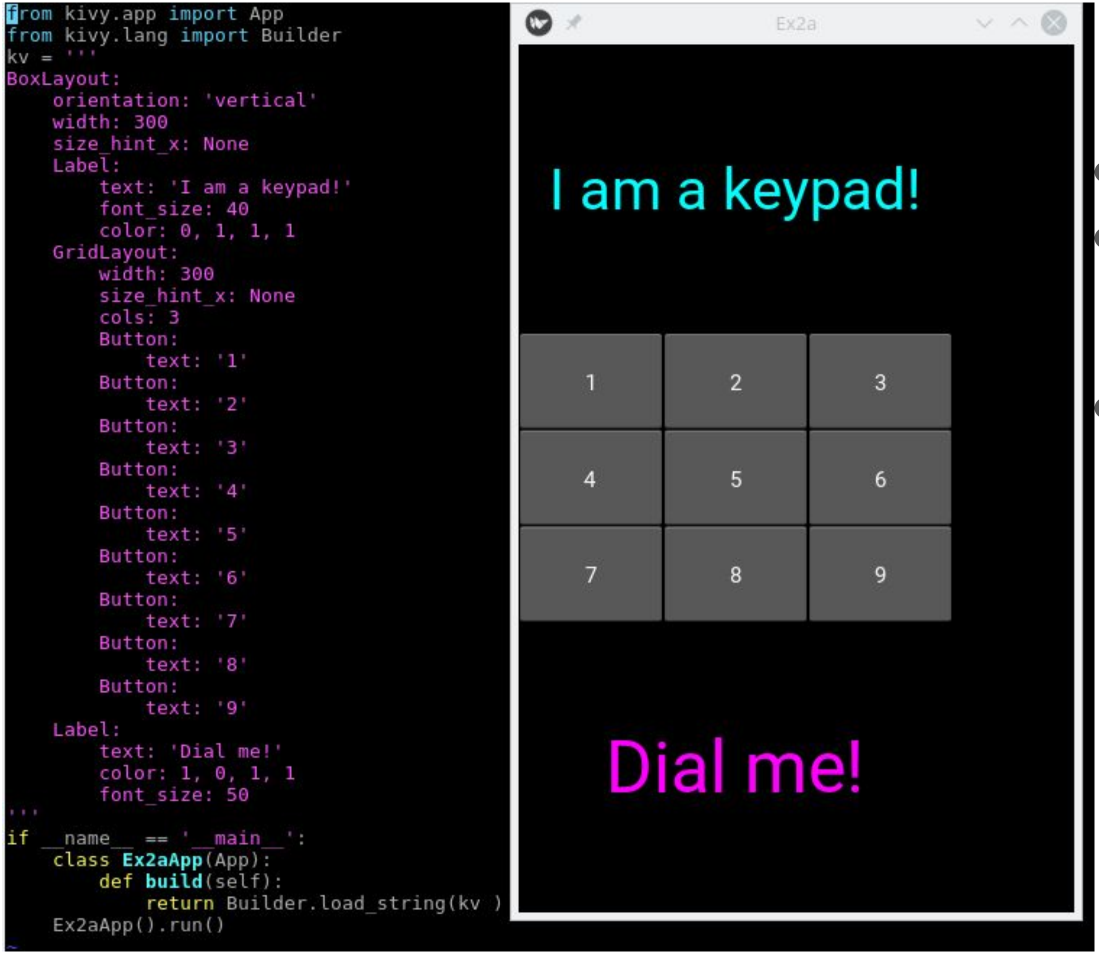
Example 2-- Layout



```
from kivy.app import App
from kivy.lang import Builder


kv = '''
BoxLayout:
    orientation: 'vertical'
    Button:
        text: 'Drink me'
    Button:
        text: 'Eat me'
    BoxLayout:
        Button:
            text: 'See me'
        Button:
            text: 'Feel me'
        Button:
            text: 'Touch me'
        Button:
            text: 'Heal me'
    Label:
        text: 'Books and Music'
'''
if __name__ == '__main__':
    class Ex2App(App):

        def build(self):
            return Builder.load_string(kv )

    Ex2App().run()
~
~
"example2.py" 30L, 555C
```

- Layout can contain widgets or other layouts
  - Just boxes here
- Complete code given
  - Execute with python3

Example 2a-- more layout



```
from kivy.app import App
from kivy.lang import Builder
kv = '''
BoxLayout:
    orientation: 'vertical'
    width: 300
    size_hint_x: None
    Label:
        text: 'I am a keypad!'
        font_size: 40
        color: 0, 1, 1, 1
    GridLayout:
        width: 300
        size_hint_x: None
        cols: 3
        Button:
            text: '1'
        Button:
            text: '2'
        Button:
            text: '3'
        Button:
            text: '4'
        Button:
            text: '5'
        Button:
            text: '6'
        Button:
            text: '7'
        Button:
            text: '8'
        Button:
            text: '9'
    Label:
        text: 'Dial me!'
        color: 1, 0, 1, 1
        font_size: 50
'''
if __name__ == '__main__':
    class Ex2aApp(App):
        def build(self):
            return Builder.load_string(kv )
    Ex2aApp().run()
```

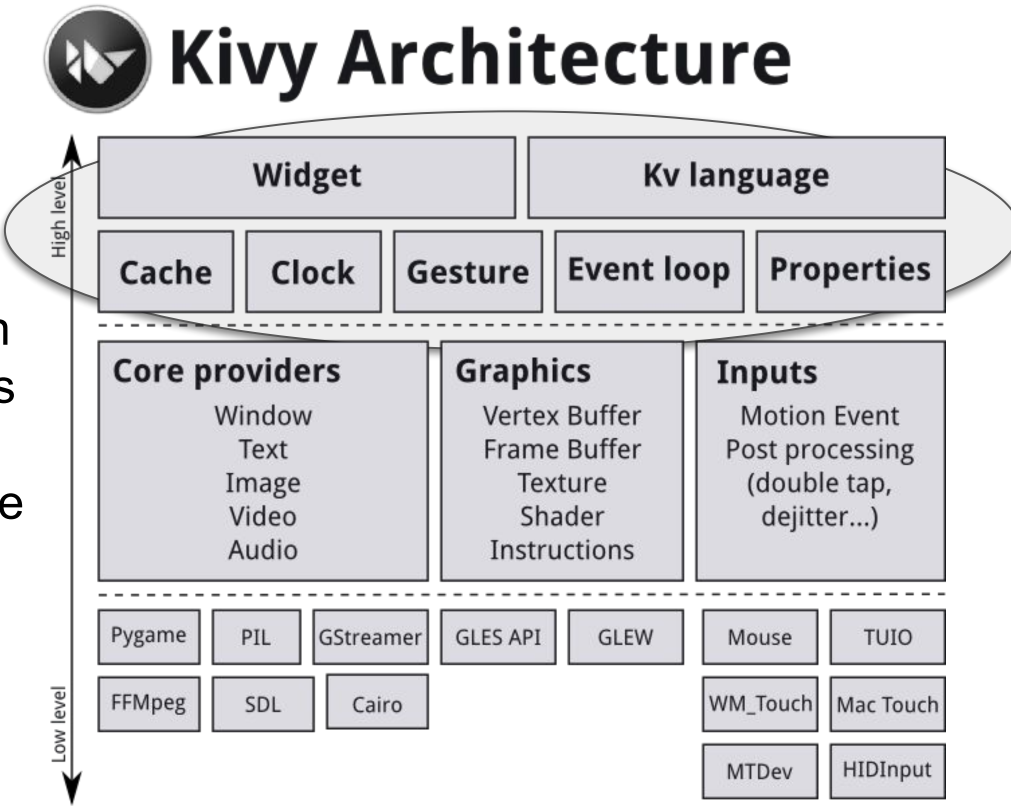
- Box layout with grid inside
- Some layouts explicitly sized
    - Child widgets are sized by parents
    - (explicit sizing must turn off hints)
- Many ways to size things
    - This is just one option

Kivy: 5 Key Concepts (cont.)

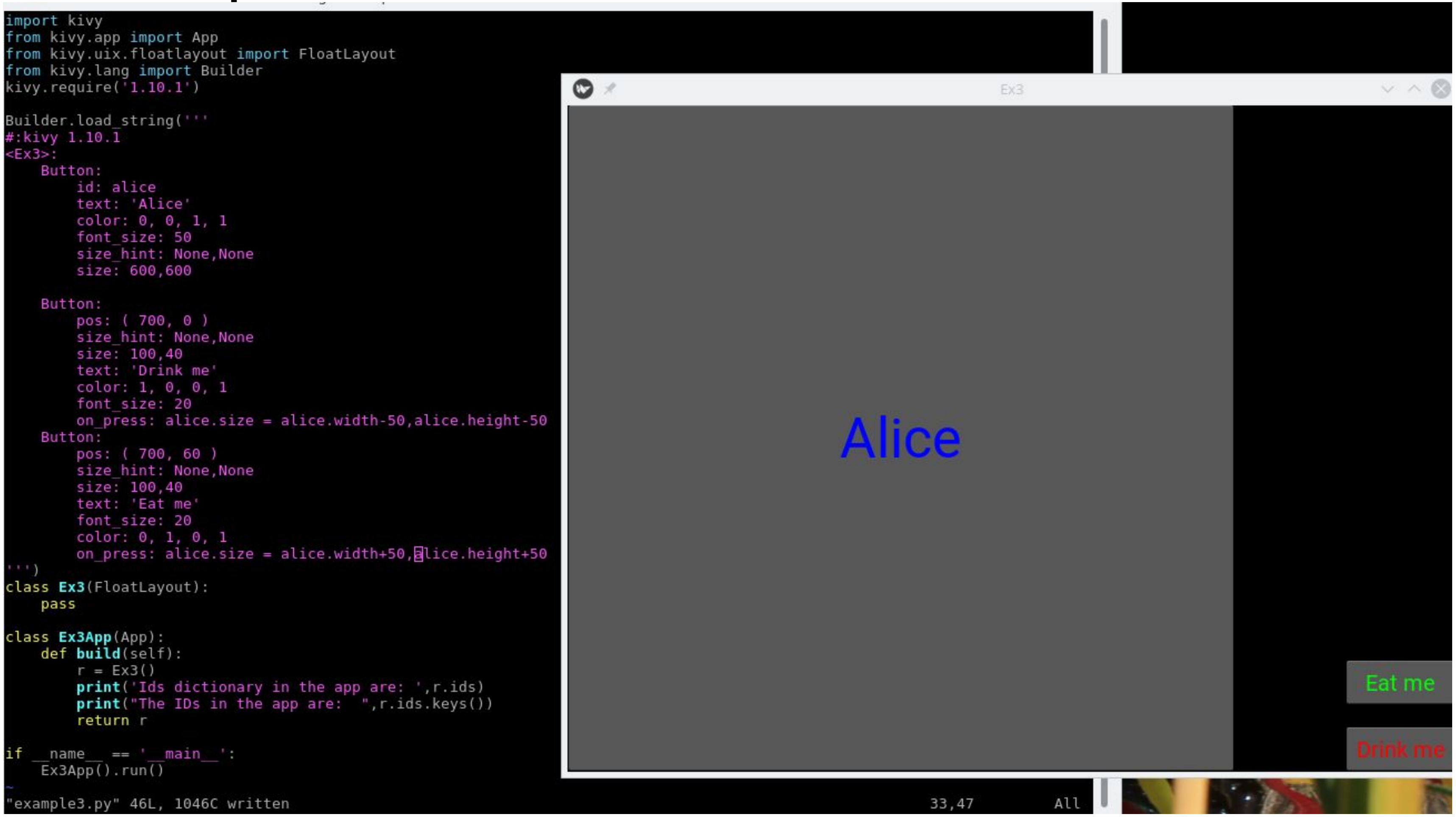
Widgets - UI elements, provides a Canvas that can be used to draw on screen. Receives and reacts to events Buttons, Labels, Switches…

Events - Widget-defined event: e.g. an event will be fired for a Button when it's pressed (on_press, on_release). Callbacks are bound to events to cause action.

Kivy Properties - produce events when an attribute changes. 'The Observer Pattern'



Example 3-- Action!



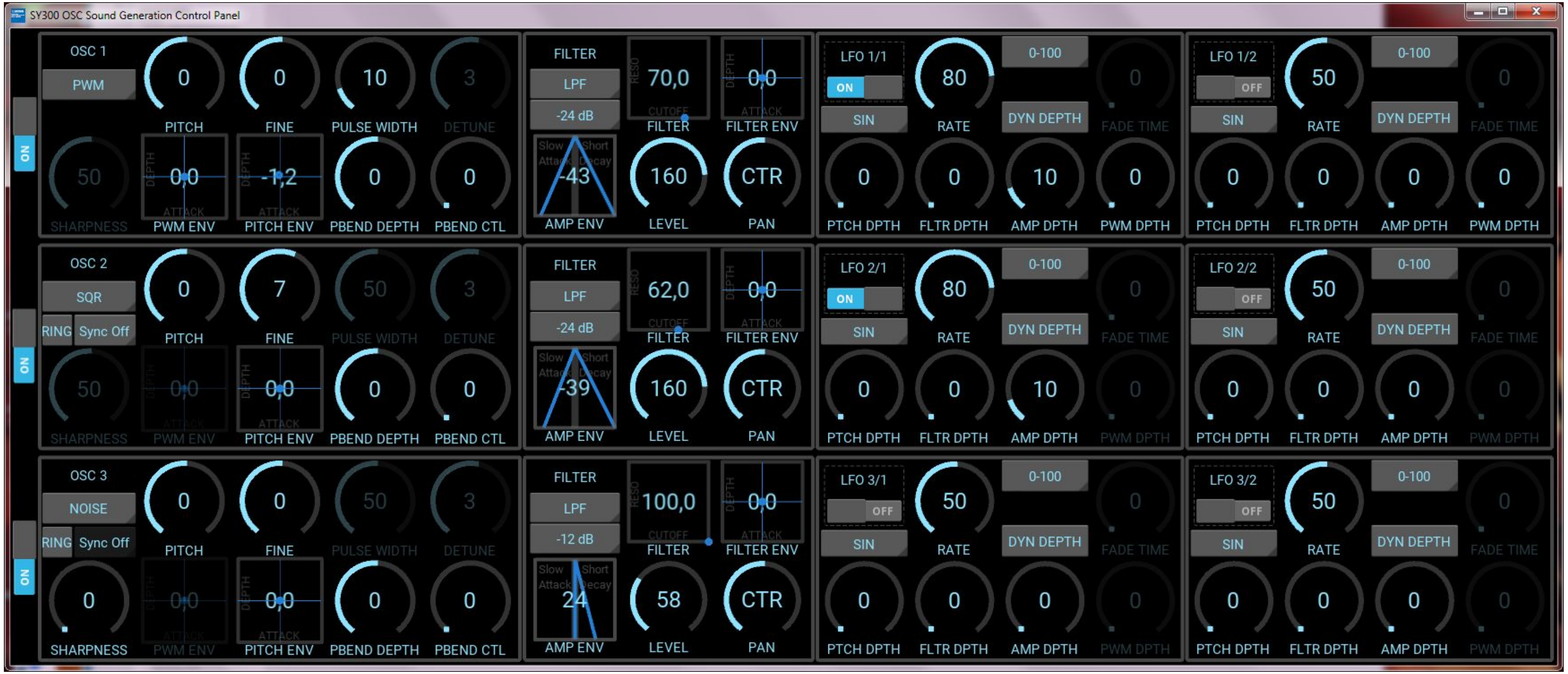
```
import kivy
from kivy.app import App
from kivy.uix.floatlayout import FloatLayout
from kivy.lang import Builder
kivy.require('1.10.1')

Builder.load_string('''
#:kivy 1.10.1
<Ex3>:
    Button:
        id: alice
        text: 'Alice'
        color: 0, 0, 1, 1
        font_size: 50
        size_hint: None,None
        size: 600,600

    Button:
        pos: ( 700, 0 )
        size_hint: None,None
        size: 100,40
        text: 'Drink me'
        color: 1, 0, 0, 1
        font_size: 20
        on_press: alice.size = alice.width-50,alice.height-50
    Button:
        pos: ( 700, 60 )
        size_hint: None,None
        size: 100,40
        text: 'Eat me'
        font_size: 20
        color: 0, 1, 0, 1
        on_press: alice.size = alice.width+50,alice.height+50
''')
class Ex3(FloatLayout):
    pass

class Ex3App(App):
    def build(self):
        r = Ex3()
        print('Ids dictionary in the app are: ',r.ids)
        print("The IDs in the app are:  ",r.ids.keys())
        return r

if __name__ == '__main__':
    Ex3App().run()
~
"example3.py" 46L, 1046C written                                                         33,47         All
```

Attaching actions

- Attach Actions to buttons
  - Specify a Python call back method
    - In Kivy language: on_button_up: MyPyCall( )
  - Specifying a kivy language consequence
    - on_button_up: self.size = (40, 100)
- Mouse motion events get passed to all widgets:
  - You get coordinates (touch), test if these collide with a specific widget

```
def on touch down(self, touch):
    if self.collide point(*touch.pos)  and not self.disabled:
        do_somthing()
```

Applying the Concepts

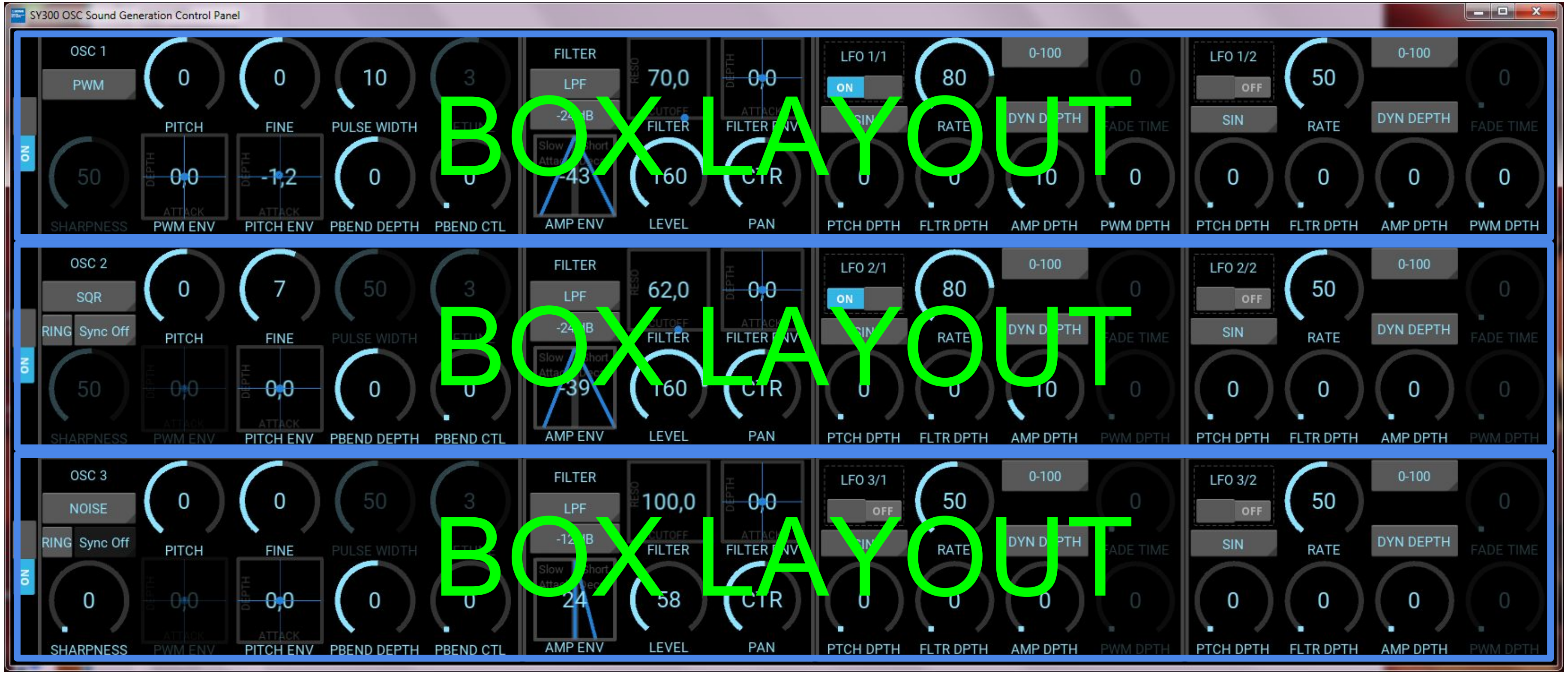

SY300 OSC Sound Generation Control Panel

ON
OSC 1: PWM
PITCH 0 · FINE 0 · PULSE WIDTH 10 · DETUNE 3
SHARPNESS 50 · PWM ENV 0,0 · PITCH ENV -1,2 · PBEND DEPTH 0 · PBEND CTL 0
FILTER: LPF · -24 dB · FILTER (RESO / CUTOFF) 70,0 · FILTER ENV (DEPTH / ATTACK) 0,0
AMP ENV (Slow Attack / Short Decay) -43 · LEVEL 160 · PAN CTR
LFO 1/1: ON · SIN · RATE 80 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 10 · PWM DPTH 0
LFO 1/2: OFF · SIN · RATE 50 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 0 · PWM DPTH 0

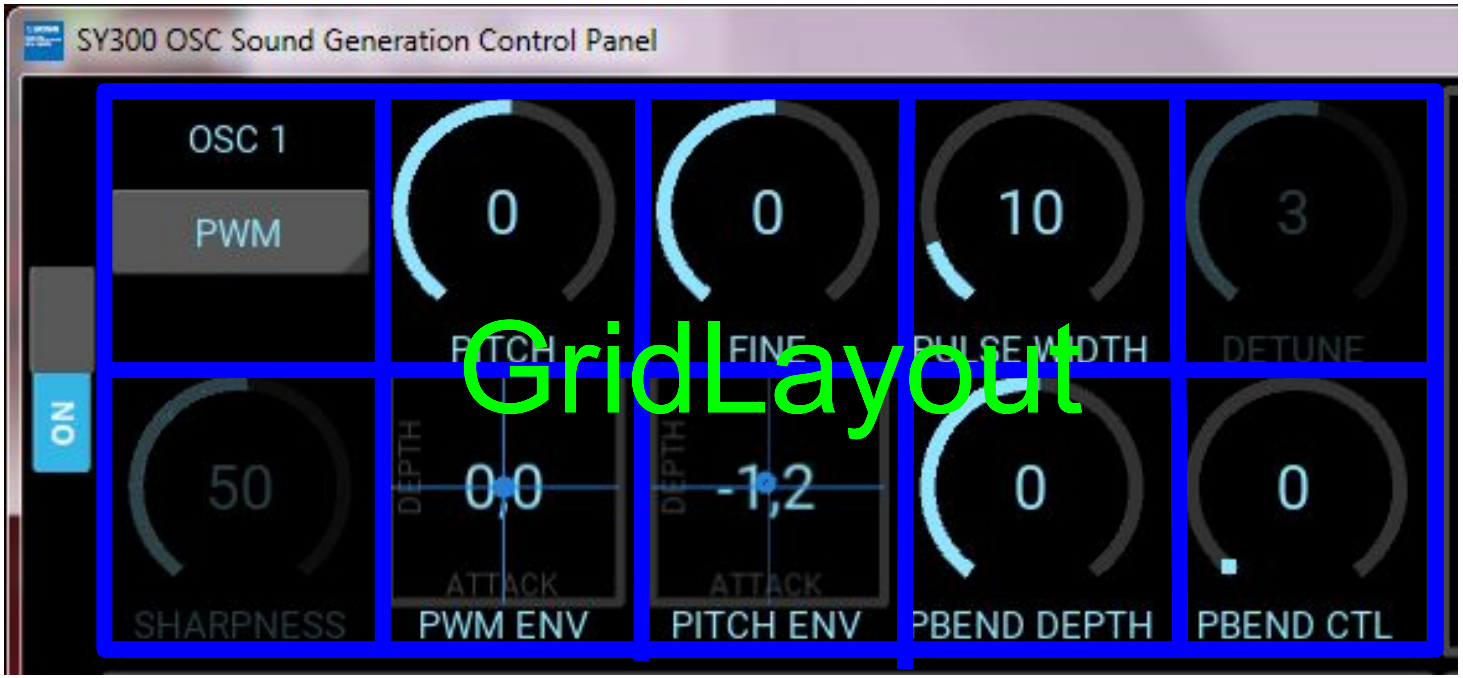
ON
OSC 2: SQR · RING · Sync Off
PITCH 0 · FINE 7 · PULSE WIDTH 50 · DETUNE 3
SHARPNESS 50 · PWM ENV 0,0 · PITCH ENV 0,0 · PBEND DEPTH 0 · PBEND CTL 0
FILTER: LPF · -24 dB · FILTER (RESO / CUTOFF) 62,0 · FILTER ENV (DEPTH / ATTACK) 0,0
AMP ENV (Slow Attack / Short Decay) -39 · LEVEL 160 · PAN CTR
LFO 2/1: ON · SIN · RATE 80 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 10 · PWM DPTH 0
LFO 2/2: OFF · SIN · RATE 50 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 0 · PWM DPTH 0

ON
OSC 3: NOISE · RING · Sync Off
PITCH 0 · FINE 0 · PULSE WIDTH 50 · DETUNE 3
SHARPNESS 0 · PWM ENV 0,0 · PITCH ENV 0,0 · PBEND DEPTH 0 · PBEND CTL 0
FILTER: LPF · -12 dB · FILTER (RESO / CUTOFF) 100,0 · FILTER ENV (DEPTH / ATTACK) 0,0
AMP ENV (Slow Attack / Short Decay) 24 · LEVEL 58 · PAN CTR
LFO 3/1: OFF · SIN · RATE 50 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 0 · PWM DPTH 0
LFO 3/2: OFF · SIN · RATE 50 · 0-100 · DYN DEPTH · FADE TIME 0
PTCH DPTH 0 · FLTR DPTH 0 · AMP DPTH 0 · PWM DPTH 0

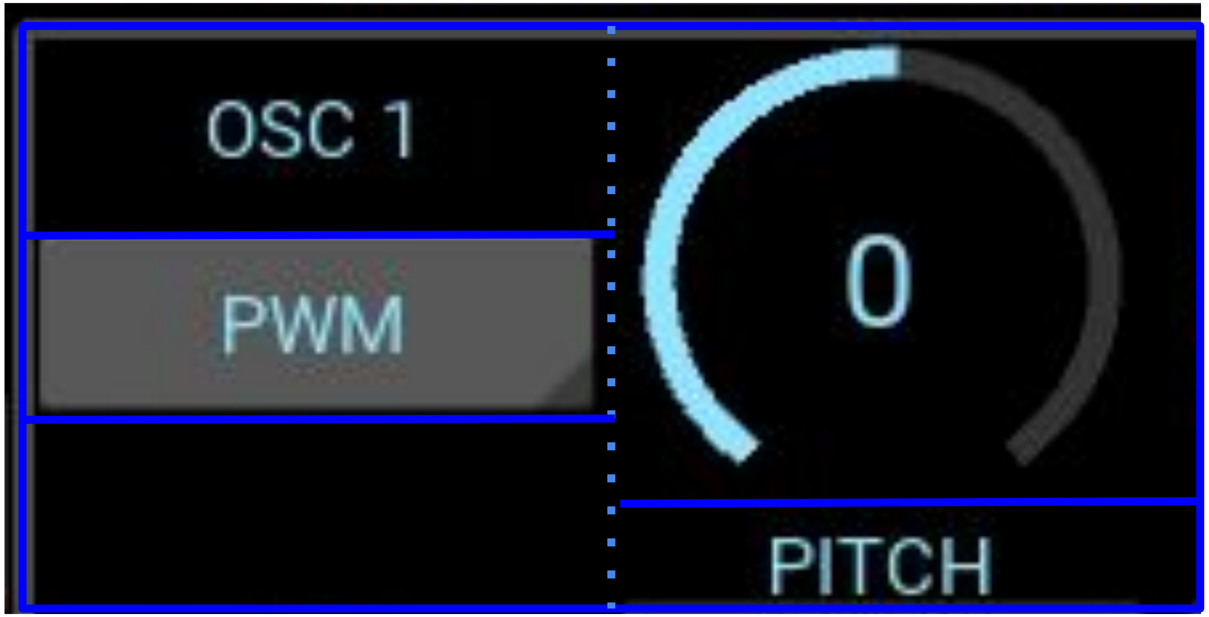
Applying the Concepts



Applying the Concepts



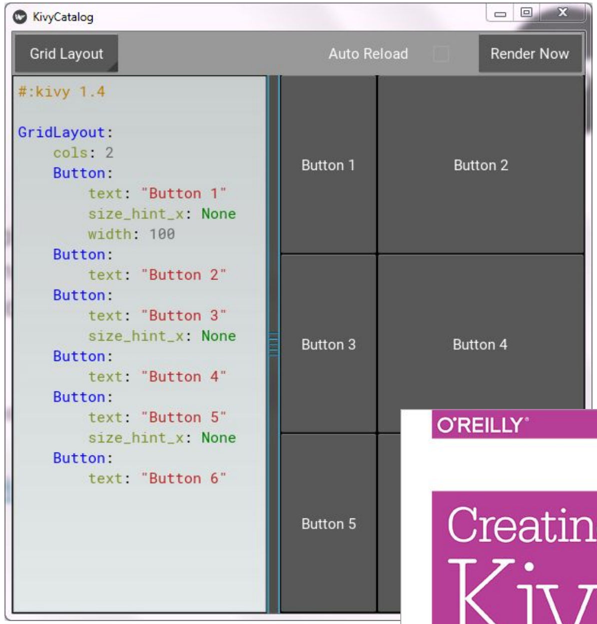
Applying the Concepts



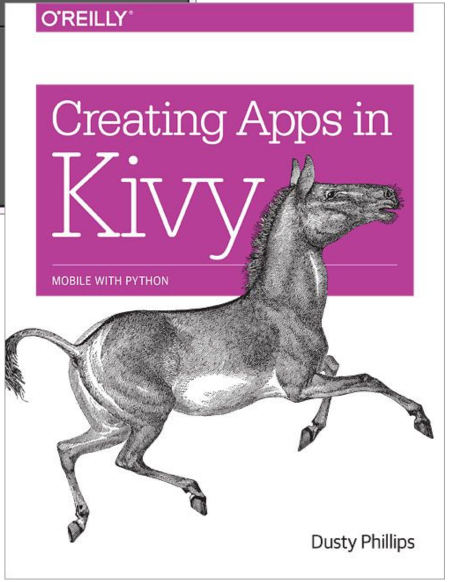
2 BoxLayouts in a GridLayout

Kivy Resources

- Kivy.org
  - Very complete and well organized
  - Excellent reference and tutorials
- Kivy Catalog, a supplied example
  - Interactive showcase of widgets and layouts
- Kivy Crash Course, Blog and videos
  - Tutorial videos
- Kivy Garden
  - User contributed codes. Many creative examples
- Kivy Source Code on GitHub
- Examples from this talk on GitHub



```
#:kivy 1.4

GridLayout:
    cols: 2
    Button:
        text: "Button 1"
        size_hint_x: None
        width: 100
    Button:
        text: "Button 2"
    Button:
        text: "Button 3"
        size_hint_x: None
    Button:
        text: "Button 4"
    Button:
        text: "Button 5"
        size_hint_x: None
    Button:
        text: "Button 6"
```



Summary

Kivy is a powerful modern GUI framework, well supported and cross platform

The **KV language** separates Interface design from app logic.

- An ‘outline’ that defines arrangement and simple behaviors.
- Makes rework and changes to organization, fast and fun
- There always seemed to be a natural place to add our logic, surprisingly easy

Great compatibility seen across Windows & Linux

PyInstaller used to create standalone executables